Lightweight 3D Human Pose Estimation Network Training Using Teacher-Student Learning

Dong-Hyun Hwang *
School of Computing
Tokyo Institute of Technology
hwang.d.ab@m.titech.ac.jp

Suntae Kim
Clova AI Video
NAVER Corp.
suntae.kim@navercorp.com

Nicolas Monet
Computer Vision Group
NAVER LABS Europe
nicolas.monet@naverlabs.com

Hideki Koike
School of Computing
Tokyo Institute of Technology
koike@acm.org

Soonmin Bae
Clova AI Video
NAVER Corp.
soonmin.bae@navercorp.com

Abstract

We present MoVNect, a lightweight deep neural network to capture 3D human pose using a single RGB camera. To improve the overall performance of the model, we apply the teacher-student learning method based knowledge distillation to 3D human pose estimation. Real-time post-processing makes the CNN output yield temporally stable 3D skeletal information, which can be used in applications directly. We implement a 3D avatar application running on mobile in real-time to demonstrate that our network achieves both high accuracy and fast inference time. Extensive evaluations show the advantages of our lightweight model with the proposed training method over previous 3D pose estimation methods on the Human3.6M dataset and mobile devices.

1. Introduction

We aim to estimate 3D human pose in real-time. Recently human pose estimation has achieved great progress and is being used for sports analytics, body and gesture motion capture in the AR (Augmented Reality) or VR (Virtual Reality) environment. As VR headset display technology becomes mature, various applications including entertainment, education, and telecommunication are getting released to the market. AR receives even more attention since AR does not require any additional equipment. Nevertheless, creating AR/VR content often requires special settings and devices. We believe that mobile-based marker-less motion capture system will accelerate the advance of AR/VR application market.

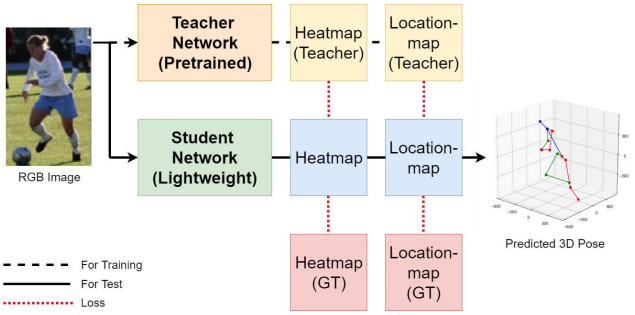

Figure 1. An overview of our proposed method. To train lightweight 3D human pose estimation model efficiently, we adopt the basis of knowledge distillation: (1) First, we train a teacher model, which consists of a large number of neural network layers. (2) Then, we train the lightweight model with extra supervision of the teacher model via mimicry loss functions for 3D pose knowledge transfer. The trained lightweight network does not depend on the teacher model and can perform efficient 3D human pose estimation.

Conventional motion capture systems are marker-based relying on wearable suits with sensors and multiple cameras or need depth cameras (e.g. Microsoft Kinect[1], Intel RealSense[2]) to obtain human joint locations. These methods usually require expensive and specialized devices or are restricted to be used in the indoor environment due to calibration procedures and specific light sources required.

Recently, by leveraging the power of deep neural networks, human pose estimation technology with RGB images has been remarkably progressed. However, perfor-

*This work was done when the first author was on an internship at Clova AI Video, NAVER Corp.

[1]https://developer.microsoft.com/en-us/windows/kinect
[2]https://software.intel.com/en-us/realsense

mance gains of deep learning-based models accompanies high deployment costs due to very deep and wide layers [40]. This leads to increased FLOPS (FLoating point OPerations per Second), which is not suitable for devices with limited computing resources such as smartphones or embedded systems. To reduce the number of FLOPS, a lightweight model is usually designed with a smaller number of parameters and with efficient operations such as depthwise convolutions. However, the significantly reduced amount of parameters affect the accuracy of the model. Methods using binarized convolutional neural network (CNN) or quantization [5, 7] often suffer from a lack of generalization capacity.

In this paper, we propose an efficient learning method for 3D human pose estimation model with minimal performance loss while reducing the number of parameters. We extend the 2D human pose estimation model learning method based on teacher-student learning [47] to 3D, and through designing and implementing MoVNect, a lightweight 3D pose estimation model. We observed that the lightweight model trained with the proposed approach achieves higher accuracy than the model trained with the vanilla method. In addition, we compare the inference time of our model with previous methods running on smartphones and develop an AR application with our model to show the effectiveness of the proposed method.

In summary, our contributions include:

- We design MoVNect, a lightweight 3D human pose estimation model that can run in real-time on hardware with limited resources such as smartphones.
- We propose a method to efficiently train lightweight 3D human pose estimation with teacher-student learning (Figure 1). The proposed method shows an accuracy improvement of 14% than the vanilla training method on the Human3.6M test set.
- The inference time of various methods on smartphones is evaluated, and the feasibility of the proposed model to be used on various hardware is verified.
- We develop a real-time mobile application of 3D avatar with our proposed model to show the practicality of our approach.

2. Related Works

In this section, we briefly discuss previous approaches for 2D and 3D single human pose estimation using a single RGB camera.

2.1. 2D Human Pose Estimation

Thanks to advances of deep neural networks, we observe huge improvements on 2D human pose estimation in recent years. Early approaches extracted features with CNN and directly estimated the joint coordinates as numerical values with fully-connected layers [22, 43].

Later, the heatmap regression-based method [6], which fully utilizes the spatial context information of the image, was proposed. In this method, a network estimates a heatmap of each joint and indicates the location of joints with the point of the maximum value of each heatmap. Since this method is more accurate than the direct regression method, it has been used in most subsequent approaches [9, 11, 12, 32, 39].

2.2. 3D Human Pose Estimation

3D human pose estimation is more challenging than 2D estimation because it is an inherently under-constrained problem that requires estimating z-axis information not included in two-dimensional images. Some early approaches used physics priors [1, 44] or semiautomatic analysis-by-synthesis fitting of parametric body models [16, 21]. 3D pose estimation has also been progressed drastically since the utilization of CNN. Most studies have attempted to solve the problem in two steps: (1) first estimate 2D joints, and then (2) lift the estimated value into 3D. There have been attempts to convert 2D joint coordinates into 3D coordinates [26, 28, 36, 41]. This method is easy to implement, however, the accuracy of 3D output highly depends on the result of the 2D joint prediction. Furthermore, because this method does not utilize the spatial information of CNN layers' outputs, the network has low generalization ability.

Recently, some researches try to combine the two steps. 3D pose estimation methods using volumetric heatmap regression were proposed [27, 34, 35]. However, volumetric heatmaps consumes a lot of memory. To address this issue, a method with stepwise depth resolution increase [35] and a soft-argmax based method to find 3D coordinates in low-resolution heatmaps [27] have been proposed.

The current state-of-the-art methods utilize multi-view geometry to train or inference the network [20, 25]. These methods achieve accurate pose estimation, however, it requires a lot of computation and memory to process multi-view images.

Another method regresses location maps where each pixel contains an estimate of a particular coordinates value [31]. Because this method uses a two-dimensional location map, it has efficient memory usage and relatively low computational cost, allowing it to run in real-time on a high-end personal computer. Since these features are suitable for our goal, our approach is designed to be based on this method.

2.3. Knowledge Distillation

Knowledge distillation is to transfer the information between different networks with distinct capacities [3, 4, 17]. The main idea of knowledge distillation is to apply extra su-

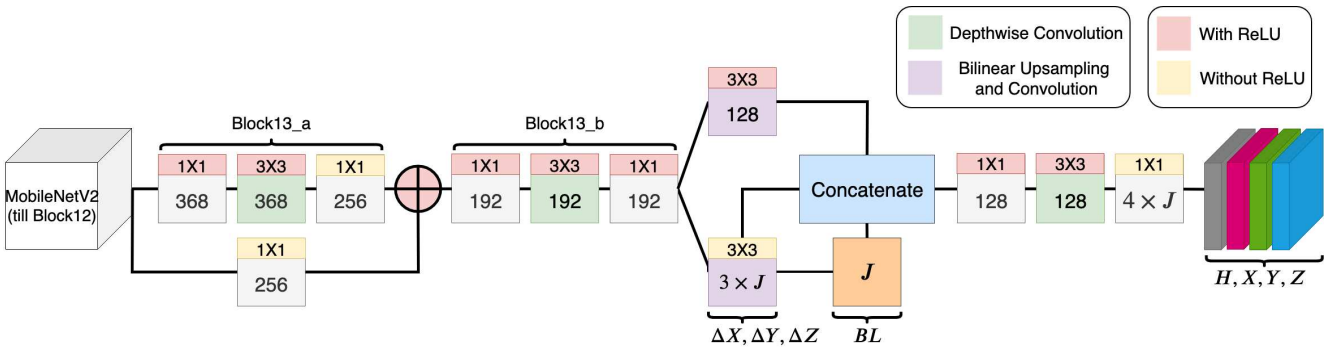

Figure 2. Network structure of MoVNect: a single RGB image is fed into the base network (MobileNetV2 till block12), and pointwise and depthwise CNN based structures are used for efficient feature extraction. The intermediate features, ΔX, ΔY, and ΔZ, are used for bone length-features, auxiliary cue to estimate root-relative 3D human pose. The network predicts heatmaps H and root-relative 3D joint location maps X, Y, Z.

pervision using the teacher model in class probabilities [17], feature representations [3, 37], or inter-layer flows [46]. It is used for efficient training of small networks difficult to train using large networks, relatively easy to train [37]. Hinton et al. successfully transferred the knowledge of a large network to a small network [17]. In addition, methods for online-based distillation [15, 48] are proposed and achieve more effective optimization than previous offline methods.

Recently, there are initial attempts to expand knowledge distillation from classification problems to human pose estimation. The initial attempt estimates human pose using radio signals [49]. In addition, a method to train an efficient lightweight 2D pose estimation model by knowledge transfer of joint heatmaps [47] is proposed and shows significant performance improvement. Although previous methods show that knowledge distillation can be applied not only to category-level discriminate knowledge but also human pose estimation [47, 49], these methods are limited to 2D human pose estimation. In this work, we propose a knowledge transfer method for 3D human pose networks using teacher-student learning. We also design MoVNect, a lightweight 3D human pose estimation network with the proposed method. Our lightweight model trained with efficient training method enables accurate pose estimation with very low computation, which can operate on devices with low processing power.

3. MoVNect: Lightweight 3D Human Pose Estimation Network

In 3D human pose estimation, we estimate the 3D pose P^{3D} from a given RGB image I. $P^{3D} \in R^{3\times J}$ represents the root-relative 3D positions of the J body joints. We assume our network runs on low power devices (e.g. smartphone, embedded system). Therefore, the network estimates 15 joints ($J = 15$), a minimum requirement for the motion of 3D full-body characters.

3.1. CNN based 3D Pose Regression Network Architecture

Among previous 3D estimation approaches, the model proposed by Mehta et al. [31] has a good balance between accuracy and inference time. It is easy to apply knowledge distillation because the location map used in the model is 2D spatial information similar to the 2D heatmap. Therefore, we design a lightweight network architecture based on Mehta et al's approach [31] with the model search procedure (session 4.3) as shown in Figure 2. Our network produces the heatmaps and location maps for all joints $j \in 1..J$. We use the till of block 12 of MobileNetV2 [38] as the base network and adopt additional depthwise CNN layers for efficient computation. We add the bone length-features to the network for an explicit clue to guide the prediction of root-relative location maps as:

$$BL_j = |\Delta X_j| + |\Delta Y_j| + |\Delta Z_j|$$

ΔX_j, ΔY_j, and ΔZ_j are intermediate features from our network. For efficient calculation, bone length-features are calculated using L1 distance instead of L2 distance-based equation proposed by Mehta et al. [31]. The calculated features are concatenated with other intermediate features and utilized to calculate the final output.

Inference: we use cropped images based on the person's bounding box when training our network. This makes our network performance affected by the size of the image at runtime. To address this issue while maintaining a real time processing on mobile devices, we acquire a bounding box based on the human keypoint K, found in the initial few frames of 2D heatmaps with a buffer area $0.2\times$ the height vertically and $0.4\times$ the width horizontally. We then track

it continuously using previous frames with a momentum of 0.75. To normalize scale, a cropped image based on the bounding box is resized to 256×256 and used as an input to the network.

3.2. Extra Supervision based on Teacher-Student Learning

A brief outline of the proposed training method is shown in Figure 1. Most previous approaches with knowledge distillation are designed for object classification with softmax cross-entropy loss [3, 17] and not suitable to transfer pose knowledge. We design mimicry loss functions for 3D pose knowledge transfer based on the method of Zhang et al. [47]. The network is trained with heatmap loss function $\mathcal{L}_{HM}$ and location map loss function $\mathcal{L}_{LM}$ as

$$\mathcal{L}_{HM} = \frac{1}{J}\sum_{j=1}^{J}\{\alpha||H_j - H_j^{GT}||_2 + (1-\alpha)||H_j - H_j^{T}||_2\}$$

$$\mathcal{L}_{LM} = \sum_{j=1}^{J}\{\alpha||H_j^{GT} \odot (L_j - L_j^{GT})||_2 + (1-\alpha)||H_j^{GT} \odot (L_j - L_j^{T})||_2\}$$

where H_j and H_j^{GT} specify the heatmaps for the jth joint predicted by the model and ground truth, respectively. $\odot$ is the Hadamard product and L_j specify the location maps for the jth joint predicted by the model . GT and T indicate ground truth and predicted results by the teacher model, respectively. α is the blending factor between the ground truth and teacher models loss terms and set to 0.5. The teacher-student learning is conducted in each mini-batch and throughout the entire training process. After the training, we only use the student model, already learned with the teachers knowledge.

3.3. Post-processing

Our model performs CNN based per-frame pose estimation, which leads to a small jitter, an unacceptable artifact in graphics applications. To reduce this temporal jittering, we apply the 1 Euro filter [8] to the predicted 2D keypoint and use the filtered keypoint K to refer to the value of the location map. The acquired 3D pose is also filtered to reduce the temporal noise of the prediction results of the continuous images.

The root-relative 3D pose acquired from the cropped image with the bounding box loses the global position information. To restore the global position P_G^{3D}, we use the following simple but effective global pose estimation equation [29]

Figure 3. 3D character control. The processed output can be easily utilized for handling a virtual avatar.

$$P_G^{3D} = \frac{\sqrt{\sum_1^J ||P_{[xy]}^j - \overline{P}_{[xy]}||_2}}{\sqrt{\sum_1^J ||K^j - \overline{K}||_2}} \begin{pmatrix} \overline{K}_{[x]} \\ \overline{K}_{[y]} \\ f \end{pmatrix} - \begin{pmatrix} \overline{P}_{[x]} \\ \overline{P}_{[y]} \\ 0 \end{pmatrix}$$

where $\overline{P}$ and $\overline{K}$ are the 3D, 2D mean over all joints. $P_{[xy]}$ is the x, y part of P^{3D} and single subscripts indicate the particular elements. f is the focal length of the camera.

Since predicted 3D pose is the root-relative 3D position of each joint, it cannot be applied directly for character animations. Hence, inverse kinematics are applied to convert the 3D position of each joint into the orientation and these orientation values are also filtered with the 1 Euro filter [8].

In addition, since our model does not have explicit knowledge of the joint angle limits of the human body, our network does not explicitly decline physically invalid poses. To address this problem, we apply the anatomical joint rotation limits to the calculated angles of each joint to ensure bio-mechanical plausibility. Through the post processing, our approach exports data directly in a format suitable to 3D character control in real-time as shown in Figure 3.

4. Experiments

4.1. Experiment Setup

We evaluate our model using two measurements:

Accuracy: To measure the accuracy of the model, we use the Human3.6M [19] dataset, currently the largest 3D pose dataset. This dataset contains 15 actions performed by 11 subjects. We employ the commonly used evaluation protocol #1: subject 1, 5, 6, 7, and 8 for training and subject

9 and 11 for testing. Mean Per Joint Position Error (MPJPE) is calculated with the root-relative 3D joint positions from our network.

Inference Time: To confirm the applicability of the proposed lightweight model in the actual mobile environment, we measure inference time on smartphone devices (Apple iPhone series) with a variety of computing hardware specifications (CPU, GPU, NPU). We use the Apple Core ML[3] framework to convert neural network models to mobile ones and to run these models on the smartphone.

4.2. Training Details

Since most 3D human pose datasets consist of indoor images only, the network, trained with only the existing 3D pose dataset, has a lack of generalizability for in-the-wild scenes. Therefore, following Mehta et al.'s method [29], we first pre-train the 2D pose estimation using LSP [23] and MPII datasets [2], and train the 3D pose estimation through Human3.6M [19] and MPI-INF-3DHP [29] datasets. Frames of 3D datasets are sampled with at least one joint movement by $>$200mm between them and cropped using the bounding box of the person. For the MPI-INF-3DHP dataset, the background augmentation is performed using the Places365 dataset [50], and finally 95k of MPI-INF-3DHP training samples and 100k of Human3.6M training samples are prepared.

We use the Keras [10] framework with the TensorFlow backend for training the network. Some random scaling (0.7-1.0) and gamma correction are performed on training. RMSProp optimization algorithm [42] with learning rate to 2.5×10^{-4} is used for 2D pose training and Adam optimization algorithm [24] with the same learning rate is used for 3D pose training. Mini-batch size is set to 4. We use the pre-trained base network with ImageNet [13] and batch normalization [18] before each non-linear activation.

4.3. Model Search

Network	Network Structure		Upsampling Method
	Block13_a	Block13_b	
Type A	368, 368, 256	192, 192, 128	Bilinear + Conv2D
Type B	368, 368, 256	192, 192, 128	TransposedConv2D
Type C	512, 512, 512	256, 256, 128	Bilinear + Conv2D

Table 1. Specification for our prototype MoVNect models. Sequential numbers on Network Structure column denote the number of CNN layers, which make up each block.

To find a suitable model, which has a good balance between accuracy and inference time, we design and train various types (Type A, B, C) of models that have a different number of layers on Block13_a, Block13_b, and upsampling method (Bilinear upsampling + Convolution, Transposed Convolution). See Table 1 for specification of our prototype MoVNect networks.

[3]https://developer.apple.com/documentation/coreml

Network	Network Structure		# Param	MPJPE
	Block13_a	Block13_b		
Type A	368, 368, 256	192, 192, 128	**1.03M**	113.3
Type C	512, 512, 512	256, 256, 128	2.69M	**108.2**

Table 2. Performance analysis with the number of layers. Metric: average MPJPE(mm). M:10^6.

First, we measure the performance and inference time correlation with the number of layers. we design MoVNect-Small (Type A), MoVNect-Large (Type C) and measure the average MPJPE on the test set of Human3.6M as shown in Table 2. Because of the deep neural network's suboptimal trade-off between the representation capability and the computational cost, Type C has no significant improvement in accuracy (about 5mm improvement), even though the number of parameters in the network is twice that of Type A.

Network	Upsampling Method	# Param	MPJPE
Type A	Bilinear + Conv2D	**1.03M**	**113.3**
Type B	TransposedConv2D	1.13M	126.7

Table 3. Performance analysis with upsampling methods. Metric: average MPJPE(mm). M:10^6

Next, we measure the change in accuracy according to the upsampling method which increases the resolution of the network output. As shown in Table 3, we compare Type A and Type B, which use bilinear upsampling with convolution and transposed convolution, respectively. According to the results, although transposed convolution method (Type B) requires more parameters, accuracy was lower than resize-convolution method (Type A). We presume that while transposed convolution method has a unique entry for each output window, resize-convolution method is implicitly weighted in a way that it reduces the high frequency artifacts. Based on these results, we finally choose Type A network for MoVNect.

5. Results

5.1. Accuracy Results on Human3.6M Dataset

Our results on Human3.6M are shown in Table 4. Our model shows competitive accuracy compared with other methods. In particular, the model trained with teacher-student learning (marked with †) shows significantly improved accuracy (14% average MPJPE reduction). Even though our model consists of a very small number of parameters, it has cost-effective accuracy. These results show that our proposed training approach has good generalization capability in yielding cost-efficient 3D pose estimation models.

We compare the computation amounts of networks in Table 5 (see column 2). Compared with the teacher model

Methods	Direct.	Discuss	Eating	Greet	Phone	Photo	Pose	Purch.	Sitting	SittingD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.	# Param
Zhou et al.[51]	87.4	109.3	87.1	103.2	116.2	143.3	106.9	99.8	124.5	199.2	107.4	118.1	114.2	79.4	97.7	113.0	-
Du et al.[14]	85.1	112.7	104.9	122.1	139.1	135.9	105.9	166.2	117.5	226.9	1120.0	117.7	137.4	99.3	106.5	126.5	-
Park et al.[33].	100.3	116.2	90.0	116.5	115.3	149.5	117.6	106.9	137.2	190.8	105.8	125.1	131.9	62.6	96.2	117.3	-
Mehta et al.[29]	52.6	63.8	55.4	62.3	71.8	52.6	72.2	86.2	120.6	66.0	79.8	64.0	48.9	76.8	53.7	68.6	-
Martinez et al.[28] w/ SH	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9	19.3M
Mehta et al.[31]	62.6	78.1	63.4	72.5	88.3	63.1	74.8	106.6	138.7	78.8	93.8	73.9	55.8	82.0	59.6	80.5	14.6M
Pavlakos et al.[34]	48.5	54.4	54.4	52.0	59.4	65.3	49.9	52.9	65.8	71.1	56.6	52.9	60.9	44.7	47.8	56.2	-
Yang et al.[45]	51.5	58.9	50.4	57.0	62.1	65.4	49.8	52.7	69.2	85.2	57.4	58.4	43.6	60.1	47.7	58.6	-
Kocabas et al.[25]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**51.8**	34M
Ours	80.6	96.3	92.2	90.4	116.1	82.1	110.9	188.4	224.6	106.9	123.2	98.9	90.4	117.3	80.5	113.3	**1.03M**
Ours†	72.4	83.4	76.9	82.1	101.9	70.4	91.8	156.5	193.0	92.8	108.4	85.1	76.8	97.2	70.5	97.3 **(14%↓)**	**1.03M**

Table 4. Results of our network's raw CNN predictions. All frames of subject 9 and 11, cropped with the ground truth bounding box, were used for evaluation. † means the model trained with the proposed teacher-student learning method. Metric: MPJPE(mm). M:10^6.

Methods	Cost-Effectiveness			Inference Time on Devices									
	MPJPE	# Param	FLOPS	iPhone7		iPhone 8		iPhone X			iPhone XS		
				CPU	GPU	CPU	GPU	CPU	GPU	NPU	CPU	GPU	NPU
Mehta et al.[31]	80.5	14.6M	7.3M	275	175	215	140	270	120	120	200	110	17
Martinez et al.[28] w/ SH	62.9	19.3M	22M	750	200	300	160	350	160	160	270	120	20
Kocabas et al.[25]	**51.8**	34M	14M	500	220	210	210	230	160	160	200	125	50
Ours	97.3	**1.03M**	**1.35M**	**48**	**56**	**40**	**33**	**37**	**28**	**28**	**32**	**22**	**6**

Table 5. Comparison of networks' cost-effectiveness and inference time on mobile devices with various hardware configurations. Metrics: average MPJPE(mm), the number of parameters, FLOPS, and average inference time(ms). M:10^6.

[31], our model only requires 7.1% (1.03M / 14.6M) parameters and 18.5% (1.35M / 7.3M) computational amount but achieves 82.7% (97.3 / 80.5) accuracy in average MPJPE. When compared with the best performer [25], our model with 3% (1.03M / 34M) parameters and 9.6% (1.35M / 14M) computational amount achieves 53.2% (97.3 / 51.8) accuracy. Our model with the proposed method has cost-effectiveness advantages over other alternative models. Note that we apply the teacher-student learning method without changing any network structure. Based on the results in Table 2, we presume that several times more parameters with additional layers are required to overcome the performance gap without our teach-student learning. This design choice is quite critical and inevitable in real-time applications.

In Figure 4, we show qualitative results on Human3.6M and MPII datasets to demonstrate the generalization of our network to general scenes.

5.2. Inference Time Benchmark Results on Mobile Devices

Table 5 (see column 3) shows the inference time benchmark results on mobile devices. Throughout all the devices we test, our models inference time outperforms other networks. Even with low-end devices (iPhone 7 with CPU), our model runs out over 20fps and with high-end devices (iPhone XS with NPU), the throughput reaches over 160fps. Note that except our model, which has low FLOPS and memory consumption, there is no other method that can perform over 10fps on CPU and GPU. Compared to Mehta et al.s model [31], which is used as the teacher model, our model performs at least 283% (iPhone XS with NPU) and up to 730% (iPhone X with CPU) faster throughput. Compared to the best performer [25], our model shows at least 393% (iPhone 7 with GPU) and up to 1042% (iPhone 7 with CPU) faster inference time.

As the device processing power increases, the difference in throughput between networks decreases. In particular, in the case of utilizing a dedicated neural network accelerator such as NPU, all models of the comparison group are able to process more than 20 fps. However, different from other networks, our network does not have a huge gap across different processing unit types. Hence, if the model inference is done by CPU in low-end devices and by NPU in high-end devices, the GPU could be fully utilized for graphic rendering, and this is a great advantage for CG applications. Furthermore, most users do not have smartphones equipped with a dedicated processor for neural networks. Our proposed approach is expected to contribute to the spread of deep learning-based interactive applications until high-end devices are deployed broadly.

5.3. Applications

Our proposed network can be applied for various interactive mobile applications because it can provide motion data in a format suitable to 3D avatar control in real-time on mobile devices. In addition, since our network has low inference time, enough times remains for CG rendering for the application.

Augmented and Virtual Reality: Smartphone, which has built-in camera, inertial measurement unit sensor, and display, is the best portable device for AR and VR appli-

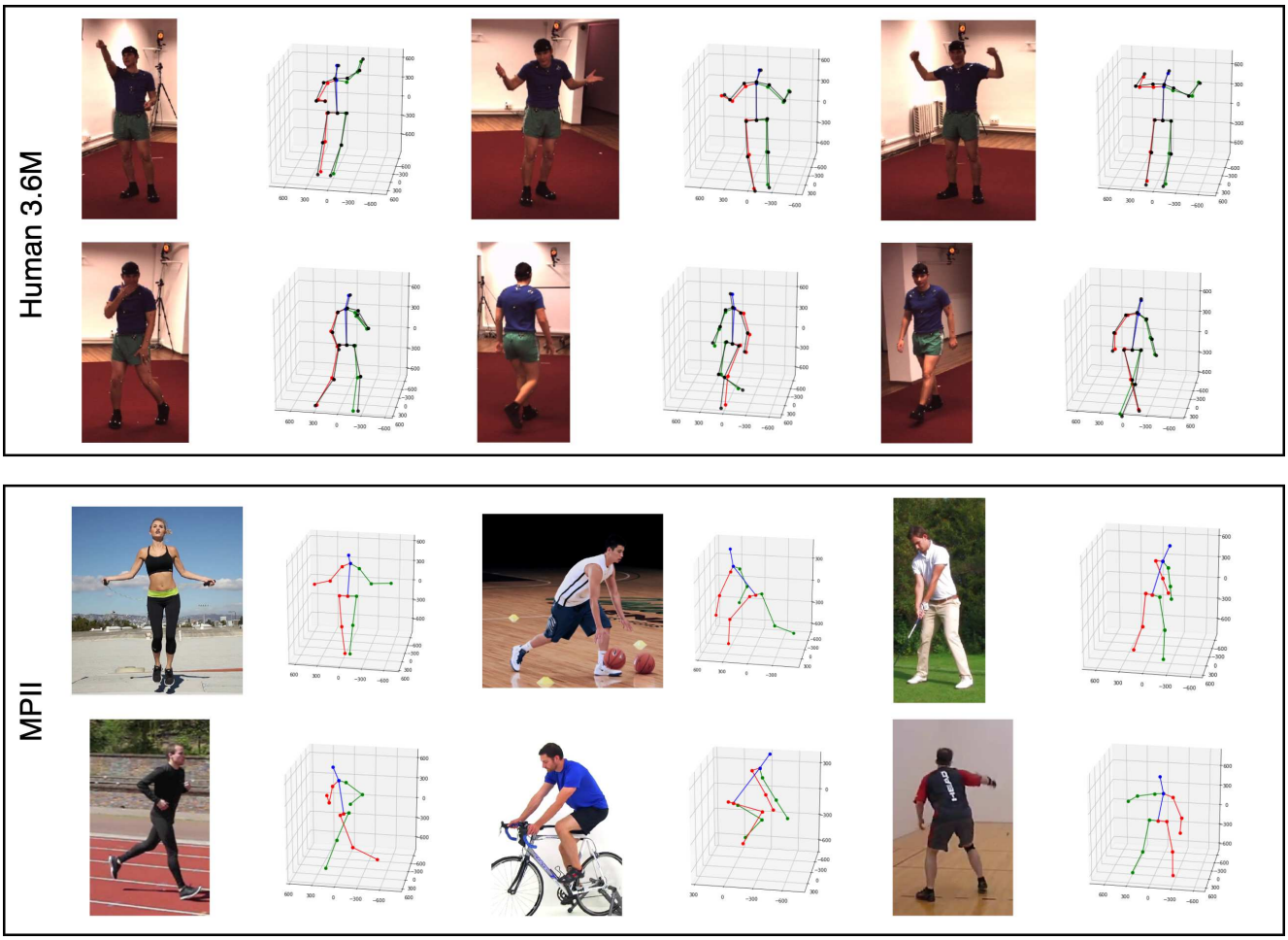

Figure 4. Qualitative results on the test set of Human3.6M(3D) and MPII(2D) datasets. Left: the input images; Right: the results of 3D pose prediction from a different viewpoint, the black skeleton is the ground truth of the Human3.6M dataset.

cations. Our method enables applications that provide the user with immersive content through a virtual avatar of the user exactly mimicking the user's real pose using a single RGB camera as shown in Figure 5. It also enables a real time interaction in body gesture capturing applications.

Motion Capture Simply Accessible: Our lightweight network can be used in a variety of devices that have low-computation power. Without communication with a high-performance server for processing algorithms, the network can be deployed and run directly on various mobile IoT devices in our daily life and can be applied in various real-life scenarios such as interactions with objects through body gestures, healthcare, and so on. For example, our algorithm can be used to recognize body language or to analyze walking postures of the elderly with common smart home devices.

6. Discussion and Future Work

To the best of our knowledge, our training approach is the first knowledge transferring method for 3D human pose estimation networks. MoVNect achieves a well-balanced performance between accuracy and inference time. Nevertheless, it still has certain limitations that can be addressed in future work. In this paper, we transfer the knowledge to the lightweight network based on the location map, thanks to the similar output type to the 2D pose network. Furthermore, because the mimicry loss function is very simple, we envision that we can easily apply our knowledge transfer method to various 3D human pose networks.

We have focused on estimating the 3D pose of a single person, which can run in real-time across various devices. Currently, latest high-end smartphones tend to be equipped with dedicated accelerators such as NPU. The proposed fully-convolutional network could be scaled to multiple persons if such devices have enough computational capacity.

Figure 5. AR-based real time 3D avatar mobile application. Our lightweight network can be utilized for interactive applications, which provide immersive experiences to users.

To reduce the resource and power consumption, most mobile deep learning frameworks do not fully support recurrent architectures. We also design our network based on per-frame prediction and this may lead to some temporal instability, similar to previous per-frame prediction approaches. We believe that our post-processing method should reduce temporal jitters enough to be usable and practical in various fields. Furthermore, in the near future, mobile devices will have more processing power and we will be able to expand per-frame to video processing levels using a recurrent approach.

In addition, to reduce inference time, our network uses a single scale of the cropped image. Processing each frame inference with multiple scales of the image (scale-space search) makes it difficult to guarantee real-time performance on low power devices. For applications that require a better accuracy for the pose, two different scales (like 0.7 and 1.0) of the cropped image can be used.

A few failure cases are illustrated in Figure 6. Our location map-based approach relies on 2D heatmap detection results and our lightweight model is not robust enough to occlusion. As future work, we will apply a pose encoding-decoding scheme [30], robust to occlusion, to our network. Despite these limitations, we observe that our method proposes an initial step in the direction of a training method for efficient lightweight 3D motion capture.

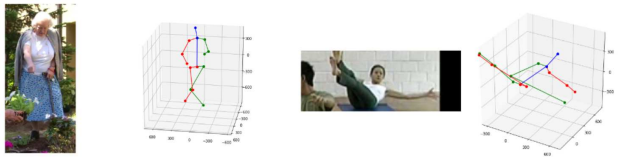

Figure 6. Failure cases of our model. Left: knees are crossed because of body part occlusion. Right: the position of the right hand is mislocated to the left hand because the right hand is occluded with the extreme pose.

7. Conclusion

In this paper, we propose MoVNect, a lightweight 3D human pose estimation model, and an efficient training strategy based on teacher-student learning. We make the step from existing 2D pose estimation with knowledge distillation to 3D pose estimation. Moreover, we present extensive evaluations on human pose and inference time benchmarks. Based on the results, we observe that our proposed teacher-student learning method significantly improves the accuracy of the model, and our network trained with the proposed method achieves very fast inference time with reasonable accuracy on various devices from low-end to high-end. We demonstrate these advantages on real mobile devices with an AR-based 3D avatar application. We hope that this work would act as an ignition of efficient training methods for lightweight neural networks in 3D human pose estimation.

References

[1] I. Akhter and M. J. Black. Pose-conditioned joint angle limits for 3d human pose reconstruction. In *IEEE CVPR 2015*.

[2] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE CVPR 2014*.

[3] J. Ba and R. Caruana. Do deep nets really need to be deep? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2654–2662. Curran Associates, Inc., 2014.

[4] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil. Model compression. In *ACM KDD 2006*, KDD '06, pages 535–541, New York, NY, USA, 2006. ACM.

[5] A. Bulat and G. Tzimiropoulos. Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources. In *IEEE ICCV 2017*.

[6] A. Bulat and G. Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *ECCV 2016*, pages 717–732.

[7] A. Bulat, G. Tzimiropoulos, J. Kossaifi, and M. Pantic. Improved training of binary networks for human pose estimation and image recognition. *arXiv preprint arXiv:1904.05868*, 2019.

[8] G. Casiez, N. Roussel, and D. Vogel. 1 filter: A simple speed-based low-pass filter for noisy input in interactive systems. In *ACM CHI 2012*, pages 2527–2530.

[9] Y. Chen, C. Shen, X.-S. Wei, L. Liu, and J. Yang. Adversarial PoseNet: A structure-aware convolutional network for human pose estimation. In *IEEE ICCV 2017*.

[10] F. Chollet et al. Keras. https://keras.io, 2015.

[11] C.-J. Chou, J.-T. Chien, and H.-T. Chen. Self adversarial training for human pose estimation. In *APSIPA ASC 2018*. IEEE.

[12] X. Chu, W. Yang, W. Ouyang, C. Ma, A. L. Yuille, and X. Wang. Multi-context attention for human pose estimation. In *IEEE CVPR 2017*.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE CVPR 2009*.

[14] Y. Du, Y. Wong, Y. Liu, F. Han, Y. Gui, Z. Wang, M. Kankanhalli, and W. Geng. Marker-less 3d human motion capture with monocular image sequence and height-maps. In *ECCV 2016*, pages 20–36.

[15] T. Fukuda, M. Suzuki, G. Kurata, S. Thomas, J. Cui, and B. Ramabhadran. Efficient knowledge distillation from an ensemble of teachers. In *Interspeech 2017*. ISCA.

[16] P. Guan, A. Weiss, A. O. Balan, and M. J. Black. Estimating human shape and pose from a single image. In *IEEE CVPR 2009*.

[17] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[18] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML 2015*, ICML'15, pages 448–456. JMLR.org.

[19] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.

[20] K. Iskakov, E. Burkov, V. Lempitsky, and Y. Malkov. Learnable triangulation of human pose. In *IEEE ICCV 2019*.

[21] A. Jain, T. Thormhlen, H.-P. Seidel, and C. Theobalt. MovieReshape. *ACM Transactions on Graphics*, 29(6):1, dec 2010.

[22] A. Jain, J. Tompson, M. Andriluka, G. W. Taylor, and C. Bregler. Learning human pose estimation features with convolutional networks. *CoRR*, abs/1312.7302, 2013.

[23] S. Johnson and M. Everingham. Clustered pose and nonlinear appearance models for human pose estimation. In *BMVC 2010*. doi:10.5244/C.24.12.

[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] M. Kocabas, S. Karagoz, and E. Akbas. Self-supervised learning of 3d human pose using multi-view geometry. In *IEEE CVPR 2019*.

[26] S. Li and A. B. Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *ACCV 2014*, pages 332–347.

[27] D. C. Luvizon, D. Picard, and H. Tabia. 2d/3d pose estimation and action recognition using multitask deep learning. In *IEEE CVPR 2018*.

[28] J. Martinez, R. Hossain, J. Romero, and J. J. Little. A simple yet effective baseline for 3d human pose estimation. In *IEEE ICCV 2017*.

[29] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt. Monocular 3d human pose estimation in the wild using improved CNN supervision. In *IEEE 3DV 2017*.

[30] D. Mehta, O. Sotnychenko, F. Mueller, W. Xu, M. Elgharib, P. Fua, H.-P. Seidel, H. Rhodin, G. Pons-Moll, and C. Theobalt. Xnect: Real-time multi-person 3d human pose estimation with a single rgb camera. *arXiv preprint arXiv:1907.00837*, 2019.

[31] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics*, 36(4), July 2017.

[32] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV 2016*, pages 483–499.

[33] S. Park, J. Hwang, and N. Kwak. 3d human pose estimation using convolutional neural networks with 2d pose information. In *Lecture Notes in Computer Science*, pages 156–169. Springer International Publishing, 2016.

[34] G. Pavlakos, X. Zhou, and K. Daniilidis. Ordinal depth supervision for 3d human pose estimation. In *IEEE CVPR 2018*.

[35] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *IEEE CVPR 2017*.

[36] D. Pavllo, C. Feichtenhofer, D. Grangier, and M. Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *IEEE CVPR 2019*.

[37] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In *ICLR 2015*, 2015.

[38] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *IEEE CVPR 2018*.

[39] K. Sun, B. Xiao, D. Liu, and J. Wang. Deep high-resolution representation learning for human pose estimation. In *IEEE CVPR 2019*.

[40] M. Tan and Q. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[41] B. Tekin, P. Marquez-Neila, M. Salzmann, and P. Fua. Learning to fuse 2d and 3d image cues for monocular body pose estimation. In *IEEE ICCV 2017*.

[42] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. *University of Toronto, Technical Report*, 2012.

[43] A. Toshev and C. Szegedy. DeepPose: Human pose estimation via deep neural networks. In *IEEE CVPR 2014*.

[44] X. Wei and J. Chai. VideoMocap. *ACM Transactions on Graphics*, 29(4):1, jul 2010.

[45] W. Yang, W. Ouyang, X. Wang, J. Ren, H. Li, and X. Wang. 3d human pose estimation in the wild by adversarial learning. In *IEEE CVPR 2018*.

[46] J. Yim, D. Joo, J. Bae, and J. Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *IEEE CVPR 2017*.

[47] F. Zhang, X. Zhu, and M. Ye. Fast human pose estimation. In *IEEE CVPR 2019*.

[48] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu. Deep mutual learning. In *IEEE CVPR 2018*.

[49] M. Zhao, T. Li, M. A. Alsheikh, Y. Tian, H. Zhao, A. Torralba, and D. Katabi. Through-wall human pose estimation using radio signals. In *IEEE CVPR 2018*.

[50] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[51] X. Zhou, M. Zhu, K. Derpanis, and K. Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *IEEE CVPR 2016*.